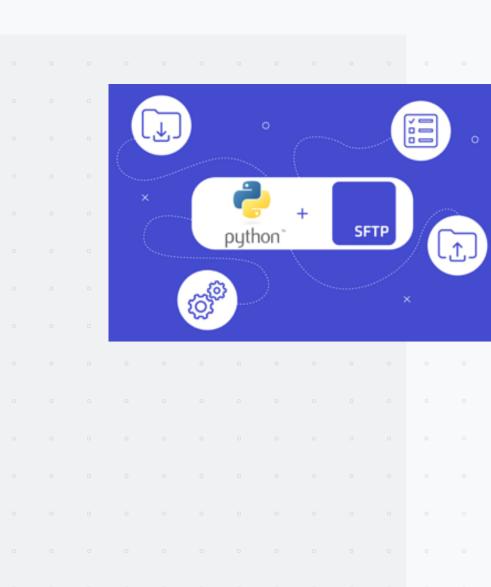


How to connect to SFTP in Python

Learn how to connect to SFTP, list files, upload and download using Python #Guides #Engineering

3 years ago by Moty Michaely - 6 min read



Python may take a tad more deciphering. With the guidance of the following post, we hope you will come out the other side feeling familiar and ready to utilize and connect to SFTP from Python. Run and Monitor Scheduled Tasks on your Favorite Apps Cron To Go simplifies the monitoring, alerting, and management of your cron jobs' performance, uptime, and status - ensuring seamless operation.

With the purpose of secure file and data transfer, SFTP has been a top

choice for many users. While it is an efficient and user friendly protocol,

engaging with an SFTP server from a programming language such as

Try Cron To Go for free! Requirements

CRON Prior to jumping in, some setup is required. We'll use the 'pysftp' package

to connect and pass commands to the SFTP server. When you are ready to install it, manually run:

Or create a requirements.txt file and declare your dependencies within. Then, save the following as requirements.txt:

```
pysftp>=0.2.9
```

1 | pip install pysftp

Then run:

```
Connecting to SFTP
```

In this post, we'll be using an environment variable named SFTPT0G0_URL

using a URI format: sftp://user:password@host. The variable is parsed to

that contains all the information required to connect to an SFTP server

Once the connection is established, the SFTP client object will be

extract the URI parts using 'urlparse', and the remote server's host key is verified by default using ~/.ssh/known_hosts.

assigned to the variable: connection.

self.connection = None

self.hostname = hostname

self.username = username self.password = password

self.port = port

def connect(self):

10

11

12

13 14

15 16

1 | pip install -r requirements.txt

import pysftp from urllib.parse import urlparse import os class Sftp: def __init__(self, hostname, username, password, port=22): """Constructor Method""" # Set connection object to None (initial value)

"""Connects to the sftp server and returns the sftp connection object""" 17 18 19 try: 20 # Get the sftp connection object 21 self.connection = pysftp.Connection(22 host=self.hostname, 23 username=self.username, 24 password=self.password, 25 port=self.port, 26 except Exception as err: 27 raise Exception(err) 28 finally: 29

```
print(f"Connected to {self.hostname} as {self.username}.")
     30
     31
     32
            def disconnect(self):
     33
               """Closes the sftp connection"""
     34
               self.connection.close()
     35
               print(f"Disconnected from host {self.hostname}")
     36
Listing Files
Now that the connection is all set up, we can use it to list files on the
remote SFTP server. This is achieved by calling the connection object's
listdir function or the listdir_attr function. Either one of these
functions can take a remote path argument or, if omitted, will list the files
and directories in the current remote directory. The listdir function
returns a list of filenames (as string), while the listdir_attr function
returns a list of SFTPAttributes objects containing the file size, creation,
modification timestamps and permissions, in addition to the file name. In
```

our example, our wrapper function returns a generator, allowing the

for obj in self.connection.listdir(remote_path):

for attr in self.connection.listdir_attr(remote_path):

The next step is to upload a file. Use the connection object's put function

and pass the path to the local file and the remote path, which is also

would look like: connection.put("./local.txt", "./remote.txt")

def upload(self, source_local_path, remote_path):

where the file should end up by the end of the upload. The function call

Uploads the source files from local to the sftp server.

f"uploading to {self.hostname} as {self.username} [(remote path:

lists all the files and directories in the specified path and returns

"""lists all the files and directories (with their attributes) in the spe

Upload File

try:

10

"./download.txt")

print(

yield attr

function caller to iterate over the returned files.

def listdir(self, remote_path):

def listdir_attr(self, remote_path):

yield obj

11 # Download file from SFTP 12 self.connection.put(source_local_path, remote_path) 13 print("upload completed") 14 except Exception as err: 15 raise Exception(err) 16 **Download File**

The last job left for us to complete is downloading our files. Use the

call the function like so: connection.get("./remote.txt",

except Exception as err:

raise Exception(err)

connection object's get function, and pass the path to the remote file

and the local path, where you would store the downloaded file. You would

```
def download(self, remote_path, target_local_path):
            Downloads the file from remote sftp server to local.
             Also, by default extracts the file to the specified target_local_path
             try:
                 print(
                     f"downloading from {self.hostname} as {self.username} [(remote pa
10
11
                # Create the target directory if it does not exist
12
                path, _ = os.path.split(target_local_path)
13
                if not os.path.isdir(path):
14
15
                     try:
                         os.makedirs(path)
16
17
                     except Exception as err:
                         raise Exception(err)
18
19
                # Download from remote sftp server to local
20
                self.connection.get(remote_path, target_local_path)
21
22
                 print("download completed")
```

So we've made it to the finish line! If you would like to run the entire program from start to finish, copy the following code and save it as main.py:

import pysftp

import os

class Sftp:

try:

finally:

def disconnect(self):

17 18 19

20

21 22

23

24 25

26 27

28

29

30 31

32

33 34

35

36 37

38 39

40

41 42

43

44 45

46 47

48 49

54

55

56 57 58 from urllib.parse import urlparse

The Whole Thing

23

24 25

def __init__(self, hostname, username, password, port=22): """Constructor Method""" # Set connection object to None (initial value) 10 self.connection = None 11 self.hostname = hostname 12 self.username = username 13 self.password = password 14 self.port = port 15 16 def connect(self):

Get the sftp connection object

username=self.username, password=self.password,

host=self.hostname,

port=self.port,

except Exception as err:

self.connection.close()

def listdir(self, remote_path):

def listdir_attr(self, remote_path):

yield obj

yield attr

try:

print(

raise Exception(err)

"""Closes the sftp connection"""

self.connection = pysftp.Connection(

print(f"Disconnected from host {self.hostname}")

for obj in self.connection.listdir(remote_path):

def download(self, remote_path, target_local_path):

for attr in self.connection.listdir_attr(remote_path):

Downloads the file from remote sftp server to local.

Create the target directory if it does not exist

"""Connects to the sftp server and returns the sftp connection object"""

print(f"Connected to {self.hostname} as {self.username}.")

"""lists all the files and directories in the specified path and returns

"""lists all the files and directories (with their attributes) in the spe

Also, by default extracts the file to the specified target_local_path

f"downloading from {self.hostname} as {self.username} [(remote pa

```
path, _ = os.path.split(target_local_path)
     59
     60
                      if not os.path.isdir(path):
     61
                          try:
     62
                              os.makedirs(path)
     63
                          except Exception as err:
     64
                              raise Exception(err)
     65
                      # Download from remote sftp server to local
     66
     67
                      self.connection.get(remote_path, target_local_path)
                      print("download completed")
     68
     69
     70
                  except Exception as err:
     71
                      raise Exception(err)
     72
     73
              def upload(self, source_local_path, remote_path):
     74
     75
                  Uploads the source files from local to the sftp server.
     76
     77
     78
                  try:
     79
                      print(
                          f"uploading to {self.hostname} as {self.username} [(remote path:
     80
     81
     82
     83
                      # Download file from SFTP
                      self.connection.put(source_local_path, remote_path)
     84
                      print("upload completed")
     85
     86
                  except Exception as err:
     87
     88
                      raise Exception(err)
     89
     90
     91
          if __name__ == "__main__":
              sftp_url = os.environ.get("SFTPTOGO_URL")
     92
     93
     94
              if not sftp_url:
     95
                  print("First, please set environment variable SFTPTOGO_URL and try again.
                  exit(0)
     96
     97
              parsed_url = urlparse(sftp_url)
     98
     99
              sftp = Sftp(
    100
                  hostname=parsed_url.hostname,
    101
    102
                  username=parsed_url.username,
    103
                  password=parsed_url.password,
    104
    105
    106
              # Connect to SFTP
    107
              sftp.connect()
    108
    109
              # Lists files with attributes of SFTP
              path = "/"
    110
    111
              print(f"List of files with attributes at location {path}:")
    112
              for file in sftp.listdir_attr(path):
                  print(file.filename, file.st_mode, file.st_size, file.st_atime, file.st_m
    113
    114
              # Upload files to SFTP location from local
    115
              local_path = "/Users/saggi/Downloads/tls2.png"
    116
              remote_path = "/tls2.png"
    117
              sftp.upload(local_path, remote_path)
    118
    119
              # Lists files of SFTP location after upload
    120
              print(f"List of files at location {path}:")
    121
              print([f for f in sftp.listdir(path)])
    122
    123
    124
              # Download files from SFTP
    125
              sftp.download(
                  remote_path, os.path.join(remote_path, local_path + '.backup')
    126
    127
              )
    128
    129
              # Disconnect from SFTP
              sftp.disconnect()
    130
Finally, run it using the command:
python main.py
Congratulations on connecting to SFTP using Python!
  Cloud FTP with maximum security and reliability
  SFTP To Go offers managed cloud storage service - highly available,
  reliable and secure. Great for companies of any size, any scale.
                                     Try SFTP To Go for free!
Check out more code samples on Github.
```

SFTP Workday

Why & How To

Laura-ann Burgess

Integration—

Do It

How to connect to SFTP in

Older post What is SFTP?

Supply Chain

Data Automation

Automation:

w. DocParser

Laura-ann Burgess

Newer post

Ruby on Rails

You might also like

Squarespace,

Laura-ann Burgess

To Go

Data Subscription &

SFTP Automation:

Make.com & SFTP

Status

Our 6 Top EDI

Laura-ann Burgess

Providers of

2024